# VisualSPARK 1.0.2 ✤ New Features!

## New Features

### Default preferences in setupcpp

- Added support for default values for the component preferences specified in a file called default.prf and residing in the working directory. The default values are used to populate the template preference file associated with the solution sequence generated by *setupcpp* for the problem.
- If the file exists upon calling *setupcpp*, the values in the file are used in place of the hard-coded default preference settings. If no such file exists, then *setupcpp* will generate a template file with the hard-coded default preferences for possible future modification by the user.

### Absolute tolerance property for each problem variable

- A new variable property, `ATOL`, was added to the *SPARK* language on the `PROBE`, `PORT` and `LINK` statements in order to specify the absolute tolerance for each problem variable. The `ATOL` property should be set to the absolute value at which the variable in question is essentially insignificant.
- In *solver*, added support to load the `ATOL` property for each problem variable, thus replacing the `AbsTolerance` keyword previously specified in the problem.prf file at the component level.
- Improved the input handler to allow it to read in properties of problem variables along with the variable values at the specified time stamps. The syntax to specify the property `ATOL` of the variable `X` is the qualified name `X:ATOL`. The input handler can read in values for any variable properties, namely `MIN`, `MAX`, `ATOL` and `INIT`.

### Sparse linear solution technique

- Added sparse linear solution method based on the C library *umfpack* (http://www.cise.ufl.edu/research/sparse/umfpack) v3.2 by Tim Davis. This new solution method is selected by specifying the value 4 for the key `MatrixSolvingMethod` in the problem.prf file. It does not rely on vendor-specific BLAS routines but instead on vanilla C code, thus ensuring portability of the *SPARK* program.
- Significant gains in calculation speed by many orders of magnitude have been observed on large problems for which the Jacobian matrix is typically more than 90% sparse.

### Full affine invariant scaling scheme

- Replaced previous scaling schemes with a single scaling scheme that achieves fully affine invariant scaling in both the variable space and the residual space. This makes *solver* less sensitive to changes in the variable units and to formulations where the variables show very different orders of magnitude.
- If convergence difficulties are encountered with a particular problem, the full scaling scheme should be selected in the problem.prf file to improve the numerical behavior of *solver*.

### Automatic Jacobian refresh strategy

- Added value 0 as a possible choice for the `TrueJacobianEvalStep` key in the problem.prf file. The value 0 indicates that the Jacobian matrix will be refreshed automatically by the solver code whenever it is needed to ensure robust and fast convergence. The refresh strategy is based on the convergence behavior of the weighted residual norm and of the increment norm between successive iterations.
- For values larger than zero, the key `TrueJacobianEvalStep` still specifies the iteration frequency at which the Jacobian should be reevaluated.

### Statistics log file

- At the end of the simulation, *solver* generates a log file with performance statistics about the solver operation. In particular, you can find information on :
- the preference settings used for the simulation for each strong component;
- the average sparsity of the Jacobian matrix and the number of times it has been refreshed; and the operation of both the linear solver and the nonlinear solver with average calculation times. This information can be used to compare the computational efficiency of solving models with different formulations or different preference settings.

### Problem driver API

- The problem driver application programming interface (API) added to *VisualSPARK* 1.0.2 allows an advanced user to:
  - customize the sequence of operations to re-solve the same problem,
  - manage and solve multiple problems,
  - retrieve solution values and specify new inputs for a new simulation of the same problem,
  - change runtime parameters (e.g., to perform sensitivity analysis).
- We have also modified the main.cpp file that implements the main *SPARK* driver function to use the new set of API. Comprehensive documentation on how to write such a problem driver function is available at http://simulationresearch.lbl.gov > VisualSPARK.

## Changes

### Documentation

- Merged glossary of terms between the *SPARK Reference Manual* and the *VisualSPARK Users Guide*.
- Enforced text conventions throughout the documentation to better distinguish the different elements of the *SPARK* language.

### Parser

- Discontinued usage of the keyword `UPDATE_FROM_LINK` in `LINK`, `PORT` and `PROBE` statements and replaced it with the keyword `INPUT_FROM_LINK` that better reflects its behavior

### Solver

***Improved convergence strategy***

- In order to ensure that badly-scaled systems are solved with satisfactorily precision, we now force the convergence check in the non-linear solver to occur in the variable space, i.e. solely

based on the increments value, after the first iteration. This represents a change from *VisualSPARK* 1.0.1 where a successful residual check carried out even after the first iteration was a sufficient condition to stop the iteration. The convergence test is stricter and more robust, especially for problems with variables with very different order of magnitudes.

***Variable scales based on individual absolute tolerances***

- The introduction of the `ATOL` property in the *SPARK* language allows to consider an individual absolute tolerance for each problem variable, as opposed to for each strong component as in *VisualSPARK* 1.0.1. The convergence check in the nonlinear solver and the computation of the cost function used with the backtracking Newton schemes have been updated to support the new absolute tolerance mechanism.

### Graphical User Interface

- Changed default wall clock settings to year 2002
- Changed title of button from "MODIFY" to "MODIFY CLASS PATH"
- Added support for new solver features in the component preferences editor
- The absolute tolerance entry has been removed from the component preferences editor because it is now specified on a variable-by variable basis.
- Added the sparse LU matrix solving method in the component preferences editor.
- Reduced scaling types to "None" and "Full" in place of the former scaling options in *VisualSPARK* 1.0.1.

## Bug Fixes

### Documentation

- Corrected typo in description of the macro links where the single quote `'` used to prefix an object name in a hierarchy of objects was replaced with the single quote `` ` `` as prescribed by the *SPARK* syntax.

  **`obj1'obj2'obj3~linkname`**

  has been replaced with

  **`obj1`obj2`obj3~linkname`**
- SPARK documentation may now be downloaded as pdf files; right-click on links below or go to http://SimulationResearch.lbl.gov > VisualSPARK.
  SPARK Reference Manual, VisualSPARK Users Guide, SPARK Library Functions, SPARK Problem Driver API.

### Parser

- Fixed a bug in the `PARAMETER` keyword (did not work when there was more than one `PARAMETER` statement in one class).
- Fixed infinite loop bug caused by aliased variables. Aliased variables are created in `LINK` statements that do NOT contain **`obj.port`** connections.

### Setupcpp

- Write out error message showing the calling syntax if *setupcpp* is executed with no argument at the command-line instead of crashing.

## Solver

- Fixed bug in *sparksym* for the *mathomatic* symbolic solver that limited to 14 the number of variables allowed in the equation to be processed.

---

SPARK was developed by the Simulation Research Group of Lawrence Berkeley National Laboratory and by Ayres Sowell Associates, with support from the Assistant Secretary for Energy Efficiency and Renewable Energy, Office of Building Technology Programs of the U.S. Department of Energy, program manager Dru Crawley.